Corruption Detection and Containment

Jeff Davis
pgsql@j-davis.com

Survey

```
ERROR:  invalid page header in
  block 123 of relation "foo"
```

Survey

- Who has seen this error?
- In production?
- Did it result in permanent data loss?
- Did you have a backup/replica?
- Was the same block bad on the backup/replica?

Who Should We Blame?

- Hardware
- Firmware
- Filesystem
- PostgreSQL
- Backup/replication policy
- ?

Who Should We Blame?

- Hardware
- Firmware
- Filesystem
- PostgreSQL
- Backup/replication policy
- ?
- (The real answer is “all of the above”, but we'll focus on these two.)

Digression: Filesystems

- In some ways, it makes sense to detect the problem in the filesystem, so that postgres and the backups don't have to worry so much
- One problem, however, is: which filesystem?
- For linux, only Btrfs and ZFS support checksums

Digression: Filesystems

- Neither filesystem has widespread use on linux.
- Both are “copy-on-write” style, meaning that every change to a block on disk writes a whole new block in a new location.
  - There are technical reasons why copy-on-write and checksums go together.

Digression: Filesystems

- But copy-on-write filesystems may not be good for database systems.
- One drawback is that you lose locality between block N and N+1, which can make sequential access into random access.
- So these filesystems might not be a suitable design for many workloads, even after they are stable and optimized

Goals

- Reliably detect corruption as early as possible
- Avoid propagating corruption (containment)
- Have a recovery plan

Goals

- All the goals are related
- If you can't detect the corruption, it will almost certainly be propagated to backups
- If you propagate corruption to backups, it compromises your recovery plan

Detection

- Page header check
- Checksums in 9.3
- pg_filedump

Detection

- Checksums in 9.3 are a big improvement over the page header check
- `initdb … --data-checksums`
- More reliable
- Detects transposed pages
- Detects corruption in the middle of a page

Detection

- `pg_filedump … | grep Error`

  is a reasonable offline check for data files
- http://pgfoundry.org/projects/pgfiledump/
- A little better than the page header check

Containment

- If corruption makes it into the executor, that causes wrong results or a mysterious crash
- If corruption makes it to the backup/replica, that eliminates recovery options

Containment

- Did you know that a base backup makes no effort to detect corruption?
- Yet we rely on it, not just for backups, but also for replication.
- Many backup/replication policies are quite likely to propagate corruption, and unlikely to offer corruption protection.
- Wait a second... backups don't protect you!?

Containment

- You really need to validate your backups.
- pg_filedump can help with this

Containment

- Streaming replication is actually quite good at avoiding the propagation of corruption, because WAL records have a CRC.
- The problem is when you have to re-sync by taking a new base backup.
- Also, you might have independent corruption on the replica that goes unnoticed.

Recovery

- Failover to good replica
- Restore from backup
- ?

Recovery

- Not much more to say here.
- If you detect corruption early enough, and prevent it from propagating, you'll be able to recover.
- Otherwise, you probably have permanent data loss.
- Perhaps with sufficient creativity, you may be able to avoid complete disaster.

Summary of the Current Situation

- Live in fear
- Use checksums in 9.3
- Make streaming replication a centerpiece of your durability strategy
- Avoid relying on frequent base backups
- Always do some verification of the base backup (pg_filedump helps here) before overwriting any previous backup

Improvements Roadmap

- http://wiki.postgresql.org/wiki/Corruption_Detection_and_Containment

Improvements Roadmap: Detection

- Detect zeroed pages as corruption
  - Right now, zero pages are always treated as valid
- Detect corruption in SLRU/clog
- Temporary files (e.g. for Sort)

Improvements Roadmap: Detection

- Distinguish between WAL corruption and end-of-WAL correctly
  - Right now, a bad CRC in a WAL record during recovery is treated as end-of-WAL
- Provide a way to enable/disable checksums

Improvements Roadmap: Containment

- Base backup should fail if corruption is detected
- More complete offline checker
- Background checker

Conclusion

- A lot to worry about
- Significant improvements being made, starting in 9.3
- Remember:
  - Streaming replication
  - checksums
  - pg_filedump
  - Be careful relying on unverified base backups